A Generic Metaheuristic Approach to Sequential Security Games

Extended Abstract

Adam Żychowski
Warsaw University of Technology
Warsaw, Poland
a.zychowski@mini.pw.edu.pl

Jacek Mańdziuk
Warsaw University of Technology
Warsaw, Poland
j.mandziuk@mini.pw.edu.pl

ABSTRACT

The paper introduces a generic approach to solving Sequential Security Games (SGs) which utilizes Evolutionary Algorithms (EAs). Formulation of the method (named EASG) is general and largely game-independent, which allows for its application to a wide range of SGs with just little adjustments addressing game specificity. Experiments performed on 3 different types of games (with 300 instances in total) demonstrate robustness and stability of EASG, manifested by repeatable achieving optimal or near-optimal solutions in the vast majority of the cases. The main advantage of EASG is time efficiency. The method scales better than state-of-the-art approaches and can be applied to sequential SGs with bigger numbers of steps compared to the existing methods. Due to anytime characteristics, EASG is very well suited for time-critical applications.

KEYWORDS
Evolutionary algorithm; Security Games; Stackelberg equilibrium

1 STACKELBERG SECURITY GAMES

We consider \( n \) step games with two players: the Defender (\( D \)) and the Attacker (\( A \)). In each time step \( \tau_1, \ldots, \tau_n \) player \( p \) chooses action \( a^p_{\tau_i} (p \in \{ D, A \}) \) from the set of available actions \( M_{\tau_i}^p \) where \( s^p_{\tau_i} \) is a state of player \( p \) in step \( \tau_i \). State \( s^p_{\tau_i} \) is determined by previous player’s actions, his initial position and the opponent’s actions. Players are not aware of the opponent’s actions. For each state \( s \) there are four predefined payoffs \( U_k^s (k \in \{ A+, A-, D+, D- \}) \) representing the Attacker’s reward (\( U^{A+} \)), their penalty (\( U^{A-} \)), the Defender’s reward (\( U^{D+} \)) and their penalty (\( U^{D-} \)), resp. Some of the states (usually those with high \( U^{A+} \) values) are distinguished as targets. If in any step \( \tau_i (i = 1, \ldots, n) \) the Attacker and the Defender move to the same state (say \( s_j \)), then the game ends (the Attacker is intercepted) and players receive payoffs \( U^{A-}(s_j) \) and \( U^{D+}(s_j) \), resp. If the Attacker reaches any of the targets (say \( s_j \)) and is not intercepted, then the game ends with the respective payoffs equal \( U^{A+}(s_j) \) and \( U^{D-}(s_j) \).

A pure strategy of a player is an assignment of one action to each potentially reachable state of the game. Let’s denote a set of all pure strategies of player \( p \) by \( \Sigma^p \). A mixed strategy \( \pi^p \) is a probability distribution over \( \Sigma^p \).

2 EVOLUTIONARY ALGORITHM FOR SGS

We consider a classical EA definition in which population of individuals of size \( p_{\text{size}} \) is maintained through generations until one of the stopping conditions is met: either the limit for generation number is reached or there is no solution improvement in a certain number of generations. Each chromosome (individual) represents some Defender’s mixed strategy (a candidate SSSE solution) in the form of a vector of pure strategies \( \pi^D \) and their respective probabilities \( p^D_1, \ldots, p^D_q \) satisfying the following equations:

\[
\pi^D = \left[ p^D_1, \ldots, p^D_q \right], \quad \sum_{i=1}^q p^D_i = 1,
\]

where \( q \) is the number of pure strategies included in the mixed strategy represented by that chromosome (\( q \) varies between individuals). A particular form of a pure strategy depends on game specificity. In the most common case, a pure strategy is represented as a list of Defender’s actions in consecutive time steps. Each chromosome in the initial is composed of one pure strategy, randomly sampled.

**Crossover.** First, a subset of \( p_{\text{crossover}} \cdot p_{\text{size}} \) individuals are randomly selected from the population, where \( p_{\text{crossover}} \) is crossover rate. Then, individuals from this subset are randomly paired and from each pair one new offspring chromosome is created in the following way. All pure strategies from the parent chromosomes are merged into one mixed strategy with their probabilities halved. Next, each pure strategy \( \pi^D \) in this newly created chromosome, except for the one with the highest probability, is removed with probability \( (1 - p^D_1)^2 \). Afterwards, probabilities of the remaining pure strategies are normalized.

**Mutation** is applied to each chromosome independently with some probability \( p_m \). First, one pure strategy in the chromosome is randomly chosen. Then iteratively, starting from a randomly selected time step \( t_i \) up to the last time step \( t_n \), an action in a
considered time step \( i_j, i \leq j \leq n \) is changed to an action uniformly chosen among all actions feasible in that state.

**Evaluation and Selection.** Chromosome fitness equals a Defender’s payoff when they play a mixed strategy encoded in that chromosome. Following [3] it is sufficient to iteratively check all pure Attacker’s strategies and select the one with the highest Attacker’s payoff (breaking ties in favor of the Defender - SStE condition). The Defender’s payoff against the above best Attacker’s response is a chromosome fitness value.

In selection, first \( e \) highest-fitted individuals (including the offsprings) are promoted. Next, a binary tournament is repeatedly executed until the next generation reaches \( p_{\text{size}} \) individuals. In the tournament two individuals are sampled with return from the current generation and the offsprings. A higher-fitted one is promoted with probability \( p_s \), otherwise a lower-fitted one is promoted.

### 3 EXPERIMENTAL EVALUATION

**Benchmark games.** Properties of EASG are tested on 3 sets of multi-step games with variable characteristic: **Warehouse Games (WHG)** [4], **Search Games (SEG)** [1], and **FlipIt Games (FIG)** [9].

In WHG there is one Defender’s unit which in a given turn can either move to an adjacent vertex or stay. Games of \( T = 3 \) – 8 steps were considered which led to game trees of \( 10^2 \) – \( 10^5 \) nodes. For each \( T \), 25 games downloaded from [8] were tested. In SEG the Defender controls several units and mobility of each of them is restricted to a subset of vertices. Furthermore, the Attacker leaves traces which makes them partially observable. In total, 90 games with \( T = 4 \) – 6, played on 3 different graph structures [1] were used. In FIG the Attacker attempts to infect certain nodes and the Defender may take actions to restore their control on the infected units. 60 FIG instances played on 3 different graph structures [2] with \( T = 3 \) – 6 were used. For each graph, 5 different payoffs structures were randomly drawn. Games were played in No-Info variant [2] (players did not know the results of their actions).

**Benchmark methods.** EASG was tested against four state-of-the-art methods for sequential general-sum extensive-form SGs BC2015 [1], C2016 [10], CBK2018 [2] and O2UCT [5, 6]. The first two are exact approaches, the remaining two yield approximate solution. BC2015, C2016 and CBK2018 are MILP-based. O2UCT relies on guided Monte-Carlo simulations.

**Experimental setup.** EASG is evaluated from three perspectives: accuracy, stability, and scalability. All results are obtained in 30 independent runs per game instance. In total EASG assessment is based on 9 000 trials (150 WHG, 90 SEG, 60 FIG, each tested 30 times) run on Intel Xeon Silver 4116 @ 2.10GHz with 256GB RAM.

**Accuracy.** A histogram of the differences between optimal and EASG solutions in all runs, across all game instances with known optimal solutions is presented in Fig. 1a. In the case of WHG, both exact methods were able to calculate the SStE for 100 games with \( 3 \) – \( 6 \) time steps. In all tests involving larger games (\( T = 7, 8 \)) the solution could not be reached due to extensive time requirements. For 72 out of these 100 games EASG obtained optimal solutions. The mean difference between EASG best results and the optimal ones was equal to 0.0013. For SEG, optimal solutions are known for 60 games with (\( T = 4, 5 \)), out of which EASG found optimal strategies in 28 cases (47%). The average divergence from the optimal results equaled 0.0253. In FIG (which are recognized as highly challenging test games, out of which EASG yielded the same solutions for 33 games). The average divergence from the optimal results equaled 0.0087.

**Stability.** Since EASG is highly non-deterministic, the ability to repeatedly reproduce good results is of paramount importance. For 45% of games, standard deviation was equal to 0. The mean standard deviation equaled 0.0059 with the maximal value 0.1629.

**Time scalability.** Figure 1 compares time efficiency of EASG vs four state-of-the-art algorithms summarized above. First, all games of a given type (separately WHG, SEG, FIG) were divided into subsets of instances with pairwise comparable game tree sizes (pairwise equal after rounding to the nearest power of 10). Then, for each subset the running times of all game instances belonging to that subset were averaged and plotted. Due to exceeding time limit of 200 hours per trial, for biggest games the results of exact methods (BC2015 and C2016) are not plotted.

**Summary.** EASG has proven to be a robust method which scales in time visibly better than state-of-the-art approaches while providing optimal or close-to-optimal solutions. It can be regarded as a MILP alternative when calculation of an exact solution is infeasible.

### ACKNOWLEDGMENTS

This work was supported by the National Science Centre, grant number 2017/25/B/ST6/02061.
REFERENCES


